UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/733,973	12/11/2003	Richard D. Dettinger	ROC920030307US1	1355

7590 10/03/2007

William J. McGinnis, Jr.
IBM Corporation
Dept. 917
3605 Highway 52 North
Rochester, MN 55901-7829

EXAMINER	
PONIKIEWSKI, TOMASZ	

ART UNIT	PAPER NUMBER
2165	

MAIL DATE	DELIVERY MODE
10/03/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/733,973
Filing Date: December 11, 2003
Appellant(s): DETTINGER ET AL.

Gero G. McClellan
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed June 14, 2007 appealing from the Office action mailed November 15 2006.

(1) Real Party in Interest

A statement identifying by name the real party of interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

The following is a listing of the evidence relied upon in the rejections of claims under appeal:

Li, U.S. Patent No. 6,748,386

Crisan et al., US Publication 2003/0191769

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 102

1. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

2. Claims 1-12, 15-28 and 35 are rejected under 35 U.S.C. 102(e) as being anticipated by Li (US 6,748,386 B1).

As per claim 1 Li is directed to a computer-implemented method of execution of a multi-step workflow that is repeatedly executed on data of a database, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format, the method comprising:

receiving current input for execution of a step of the workflow on relevant data of the database, wherein the step has been previously executed on the relevant data using

previous input identical to the current input and wherein the previous execution of the step produced previous output (column 6, lines 55-56; column 7, lines 55-61);

determining whether the step is deterministic, whereby the step generates identical output for given input in repeated executions of the step on the relevant data (column 7, lines 54-61, wherein the cached results would not be retrieved if the query wasn't the same);

and if the step is deterministic, returning the previous output produced during the previous execution of the step without re-executing the step (column 7, lines 55-61).

As per claim 2 Li is directed to comprising using the returned previous output as input to a next sequential step in the workflow (column 7, lines 54-61, wherein saved and retrieved once may mean it could be used again for next step in the process).

As per claim 3 Li is directed to determining whether the step is deterministic comprises determining whether a workflow description of the step includes a deterministic flag indicating that the step generates identical output for given input in repeated executions of the step on the relevant data (column 7, lines 54-61, wherein since the saved result is retrieved that means there is some sort of indicator like a flag to show that the output is the same).

As per claim 4 Li is directed to determining whether the current input and the previous input are the same (column 7, lines 57-61, wherein since the saved result is retrieved means that there was a process of determination);

and returning the previous output produced during the previous execution of the step only if the current input and the previous input are determined to be the same (column 7, lines 57-61).

As per claim 5 Li is directed to determining whether the current input and the previous input are the same comprises accessing a hash table representative of the previous input (column 7, lines 52-53, wherein has table could be cache).

As per claim 6 Li is directed to determining whether the relevant data has been changed since the previous execution (column 7, lines 25-28);

and returning the previous output produced during the previous execution of the step only if the relevant data has not been changed (column 7, lines 57-61).

As per claim 7 Li is directed to determining whether the relevant data has been changed comprises:

determining a timestamp indicating a point of time of the previous execution (column 7, lines 53-55);

and determining, from a transaction log of the database, whether transactions relative to the relevant data have occurred since the point of time indicated by the timestamp (column 7, lines 61-66).

As per claim 8 Li is directed to comprising:

if the relevant data has been changed since the previous execution:

executing the step on the relevant data to obtain a result (column 6, lines 55-57);

and storing the result as output to be returned for subsequent invocations of the step taking input identical to the current input, in which case execution of the step is avoided and the stored output is returned for the step (column 7, lines 7-9).

As per claim 9 Li is directed to wherein the current input comprises one or more result fields and input parameters (column 6, lines 13-14, wherein "input parameters" could be multiple "parameters").

As per claim 10 Li is directed to a computer-implemented method for managing execution of a workflow that is repeatedly executed on data of a database, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format, the method comprising:

receiving current input for execution of a step of the workflow on relevant data of the database, (column 7 lines 3-7);

identifying the step as deterministic, wherein the step generates identical output for given input in repeated executions of the step on the relevant data (column 7, lines 56-57, wherein if the result check of the query is cached then it means that another query with identical input has been executed prior to this instance)

upon determining that the step had been previously executed using input identical to the current input (column 7, lines 25-24):

returning output obtained in the previous execution of the step using input identical to the current input without executing the step using the received current input (column 7, lines 52-61); and

upon determining that the step has not been previously executed using input identical to the current input:

executing the step for the current input on the relevant data to obtain a result (column 7, lines 17-21); and

storing the result to enable managing a next invocation of the step in which the step is passed input identical to the current input, in which case the stored result is returned as output for the step without re-executing the step (column 7, lines 21-24).

As per claim 11 Li is directed to further comprising, upon determining that the step has been previously executed using input identical to the current input and prior to returning the output:

determining whether the relevant data has been changed since the previous execution of the step using the input identical to the current input (column 7, lines 25-28); and

if the relevant data has not been changed, retrieving the output obtained in the previous execution of the step using the input identical to the current input (column 7, lines 57-59).

As per claim 12 Li is directed to determining whether the relevant data has been changed comprises:

determining a timestamp indicating a point of time of the previous execution (column 7, lines 54-57);

and determining, from a transaction log of the database, whether transactions relative to the relevant data have occurred since the point of time indicated by the timestamp (column 7, lines 62-66).

As per claim 15 Li is directed to a computer readable storage medium containing a program which, when executed by a processor, performs an operation of managing execution of a multi-step workflow that is repeatedly executed on data of a database, wherein the workflow is defined by plurality of steps, each step being an executable

function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format, the operation comprising:

receiving current input for execution of a step of the workflow on relevant data of the database, wherein the step has been previously executed on the relevant data using previous input identical to the current input and wherein the previous execution of the step produced previous output (column 6, lines 55-56; column 7, lines 55-61);

determining whether the step is deterministic, whereby the step generates identical output for given input in repeated executions of the step on the relevant data (column 7, lines 54-61, wherein the cached results would not be retrieved if the query wasn't the same);

and if the step is deterministic, returning the previous output produced during the previous execution of the step without re-executing the step (column 7, lines 55-61).

As per claim 16 Li is directed to the operation further comprises: inputting the returned previous output to a next sequential step in the workflow (column 7, lines 55-61).

As per claim 17 Li is directed to determining whether the step is deterministic comprises determining whether a workflow description of the step includes a deterministic flag indicating that the step generates identical output for given input in repeated executions of the step on the relevant data (column 7, lines 54-61, wherein

since the saved result is retrieved that means there is some sort of indicator like a flag to show that the output is the same).

As per claim 18 Li is directed to the operation further comprises:

determining whether the current input and the previous input are the same (column 7, lines 57-61, wherein since the saved result is retrieved means that there was a process of determination);

and returning the previous output produced during the previous execution of the step only if the current input and the previous input are determined to be the same (column 7, lines 57-61).

As per claim 19 Li is directed to determining whether the current input and the previous input are the same comprises accessing a hash table representative of the previous input (column 7, lines 52-53, wherein has table could be cache).

As per claim 20 Li is directed to the operation further comprises:

determining whether the relevant data has been changed since the previous execution (column 7, lines 25-28);

and returning the previous output produced during the previous execution of the step only if the relevant data has not been changed (column 7, lines 57-61).

As per claim 21 Li is directed to determining whether the relevant data has been changed comprises:

retrieving a timestamp indicating a point of time of the previous execution (column 7, lines 53-55);

and retrieving a transaction log of the database (column 7, line 59, wherein the log is the query holding the query information);

and determining, from the transaction log, whether transactions relative to the relevant data have occurred since the point of time indicated by the timestamp (column 7, lines 61-66).

As per claim 22 Li is directed to the operation further comprises:

if the relevant data has been changed since the previous execution:

executing the step on the relevant data to obtain a result (column 6, lines 55-57);

and storing the result as the output to be returned for subsequent invocations of the step taking input identical to the current input, in which case execution of the step is avoided and the stored output is returned for the step (column 7, lines 7-9).

As per claim 23 Li is directed to the current input comprises one or more result fields and input parameters (column 6, lines 13-14, wherein "input parameters" could be multiple "parameters").

As per claim 24 Li is directed to a computer readable storage medium containing a program which, when executed by processor, performs an operation of managing execution of a workflow that is repeatedly executed on data of a database, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format, the operation comprising:

receiving current input for execution of a step of the workflow on relevant data of the database, wherein the step generates identical output for given input in repeated executions of the step on the relevant data (column 6, lines 64-67; column 7, lines 1-7);

and without executing the step using the current input, returning output obtained in a previous execution of the step using input identical to the current input (column 7, lines 52-61).

As per claim 25 Li is directed to the operation further comprises, prior to returning the output:

determining whether the step has been previously executed using the input identical to the current input (column 7, lines 3-7, wherein the wrapper can then check if identical query has been executed);

if so, determining whether the relevant data has been changed since the previous execution of the step using the input identical to the current input (column 7, lines 25-28);

and if the relevant data has not been changed, retrieving the output obtained in the previous execution of the step using the input identical to the current input (column 7, lines 57-59).

As per claim 26 Li is directed to determining whether the relevant data has been changed comprises:

retrieving a timestamp indicating a point of time of the previous execution (column 7, lines 53-55);

retrieving a transaction log of the database (column 7, line 59, wherein the log is the query holding the query information);

and determining, from the transaction log, whether transactions relative to the relevant data have occurred since the point of time indicated by the timestamp (column 7, lines 61-66).

As per claim 27 Li is directed to the operation further comprises:

if the step has not been executed using the input identical to the current input (column 7, lines 57-59, wherein the cache has no instance of stored transaction):

executing the step for the current input on the relevant data to obtain a result (column 6, lines 64-67; column 7, lines 1-3);

and storing the result to enable managing a next invocation of the step in which the step is passed input identical to the current input, in which case the

stored result is returned as output for the step without re-executing the step (column 7, lines 57-61).

As per claim 28 Li is directed to the operation further comprises:

if the relevant data has been changed since the previous execution of the step using the input identical to the current input (column 7, lines 62-67; column 8, lines 1-4):

executing the step for the current input on the relevant data to obtain a result (column 6, lines 64-67; column 7, lines 1-3);

and storing the result to enable managing a next invocation of the step in which the step is passed input identical to the current input, in which case the stored result is returned as output for the step without re-executing the step (column 7, lines 57-61).

As per claim 35 LI is directed to a computer-implemented method of managing execution of a workflow that is repeatedly executed on data of a database, wherein the workflow is defined by plurality of steps, each being an executable function that operates on input from a previous step and produces output for subsequent step and each step having a defined input format and output format, the method comprising:

receiving current input for execution of a step of the workflow on relevant data of the database (column 6, lines 55-56; column 7, lines 55-61);

identifying the step as deterministic, whereby the step generates identical output for given input in repeated executions of the step on the relevant data (column 7, lines

56-57, wherein if the result check of the query is cached then it means that another query with identical input has been executed prior to this instance);

upon determining that the step had been previously executed using input identical to the current input, determining whether the relevant data has been changed since the previous execution of the step using the input identical to the current input (column 7, lines 25-24);

if the relevant data has been changed since the previous execution off the step using the input identical to the current input:

executing the step for the current input on the relevant data to obtain a result (column 7, lines 17-21); and

storing the result to enable managing a next invocation of the step in which the step is passed input identical to the current input, in which case the stored result is returned as output for the step without re-executing the step (column 7, lines 21-24); and

if the relevant data has not been changed since the previous execution of the step using the input identical to the current input:

returning output obtained in the previous execution of the step using the input identical to the current input without executing the step using the received current input (column 7, lines 52-61).

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 29-30, 32-34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Li (US 6,748,386 B1) in view of Crisan et al. (US 2003/0191769 A1).

As per claim 29 Li is directed to a computer system, comprising:

a database having data (column 3, lines 50-52);

the workflow execution manager being configured for:

receiving current input for execution of a step of the workflow on relevant data of the database, wherein the step has been previously executed on the relevant data using previous input identical to the current input and wherein the previous execution of the step produced previous output (column 6, lines 55-56; column 7, lines 55-61);

determining whether the step is deterministic, whereby the step generates identical output for given input in repeated executions of the step on the relevant data (column 7, lines 54-61, wherein the cached results would not be retrieved if the query wasn't the same);

and if the step is deterministic, returning the previous output produced

during the previous execution of the step without re-executing the step (column 7, lines 55-61).

Li does not teach a workflow execution manager residing in memory for managing execution of a multi-step workflow that is repeatedly executed on the data of the database, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format.

Crisan et al. teaches a workflow execution manager residing in memory for managing execution of a multi-step workflow that is repeatedly executed on the data of the database, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format (Crisan et al., paragraph 0130, lines 4-15)

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the Li teachings of Crisan et al. to include providing an interface for specifying a single multi-analysis functional module used to execute the plurality of functional modules, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format because an output of one operation could be useful as an input for another.

As per claim 30 Li is directed to a computer system, comprising:

a database having data (column 3, lines 50-52);

the workflow execution manager being configured for

receiving current input for execution of a step of the workflow on relevant data of the database, wherein the step generates identical output for given input in repeated executions of the step on the relevant data (column 6, lines 55-56; column 7, lines 55-61);

and without executing the step using the current input, returning output obtained in a previous execution of the step using input identical to the current input (column 7, lines 57-61).

Li does not teach a workflow execution manager residing in memory for managing execution of a workflow that is repeatedly executed on the data of the database, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format.

Crisan et al. teaches a workflow execution manager residing in memory for managing execution of a workflow that is repeatedly executed on the data of the database, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format (Crisan et al., paragraph 0130, lines 4-15)

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the Li teachings of Crisan et al. to include providing an interface for specifying a single multi-analysis functional module used to execute the plurality of functional modules, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format because an output of one operation could be useful as an input for another.

As per claim 32 Li is directed to a computer-implemented method of automatically executing a plurality of functional modules from within an application, comprising:

receiving current input for execution of at least one of the functional modules, wherein the at least one functional module has been previously executed using previous input identical to the current input (Li, column 6, lines 64-67; column 7, lines 1-7; column 7, lines 57-61);

determining whether the at least one functional module is deterministic, whereby the at least one functional module generates identical output for given input in repeated executions of the at least one functional module (Li, column 7, lines 54-61, wherein the cached results would not be retrieved if the query wasn't the same);

and if the at least one functional module is deterministic, returning previous output produced during the previous execution without re-executing the at least one functional module (Li, column 7, lines 55-61).

Li does not teach providing an interface for specifying a single multi-analysis functional module used to execute the plurality of functional modules, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format.

Crisan et al. teaches providing an interface for specifying a single multi-analysis functional module used to execute the plurality of functional modules, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format (Crisan et al., paragraph 0130, lines 4-15).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the Li teachings of Crisan et al. to include providing an interface for specifying a single multi-analysis functional module used to execute the plurality of functional modules, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format because an output of one operation could be useful as an input for another.

As per claim 33 Li as modified is directed to comprising retrieving information regarding execution of the plurality of functional modules from a configuration file (Li, column 7, line 59, wherein "plurality of functional modules" means the "results" and "configuration file" means "cache").

As per claim 34 Li as modified is directed determining whether the at least one functional module is deterministic comprises examining information regarding the at least one functional module retrieved from the configuration file (Li, column 7, lines 61-66).

(10) Response to Argument

Appellants' argument regarding claims 1, 10, 15 and 35, that Li is not directed to "workflow" is not found persuasive.

Li teaches submitting requests for a search and saving the results in a cache. If the same search is requested subsequently then the search result is retrieved from the cache without the need to run the search again. Appellant argues that the word "workflow" is not present in the prior art. The definition of workflow is many, but on the general level it is a set of instruction followed in sequence wherein the output of one instruction may be used in the next instruction. Li teaches, in column 7, lines 57-61, submitting requests one after another. Li also teaches reusing output when subsequent

query is determined to be the same by application server. Therefore the examiner interprets the teachings of Li to overcome the limitations of the present application.

Appellants argument regarding claims 1, 10, 15 and 35, that Li does not teach "determining whether the step is deterministic, whereby the step generates identical output for given input in repeated executions of the step on the relevant data, and if the step is deterministic, returning the previous output produced during the previous execution of the step without re-executing the step" is not found persuasive.

Li teaches, in column 7, lines 53-61, that a timestamp may be useful to analyze web sites. Li also teaches storing results of a request in a cache. The result is retrieved for subsequent request without repeating the search unless a data change is detected which could invalidate the saved data in the cache. When the change is detected the search is performed to update the result.

Appellants' argument regarding claim 24 that Li does not "generate identical output for given input in repeated executions of the step on a relevant data" is not found persuasive.

Li teaches as previously noted in column 7, lines 53-66, that a timestamp may be used to analyze websites. Li also teaches storing results of a given request in a cache. Unless data change is detected the results stored in the cache are identical for the subsequent query. The results in the cache cannot be changed; therefore the results are identical for the subsequent step.

Appellants' argument regarding claims 29-30 and 32 although rejected under 103, appellants' argument specifies that Li does not teach "receiving current input to a step of the workflow on relevant data of the database, wherein the step has been previously executed on the relevant data using previous input identical to the current input and wherein the previous execution of the step produced previous output" and "...determining whether the step is deterministic, in that the step generates identical output for given input in repeated executions of the step on the relevant data" is not found persuasive.

As stated above Li teaches submittal of requests and also shows that subsequent requests are being made in column 7, lines 57-59. Li also shows that results of one request are being stored in a cache and then retrieved if it is determined that they are requested for subsequent request without rerunning the execution of the subsequent request.

Appellants' note that the motivation for combining teachings of Li in view of Crisan et al. for claims 29 and 30 is a mistype. The statements preceding the motivation describe the correct limitation that is being addressed. The correct motivation for claim 29 should have been:

"It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the Li teachings of Crisan et al. to include a workflow execution manager residing in memory for managing execution of a multi-step workflow

that is repeatedly executed on the data of the database, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format because an output of one operation could be useful as an input for another. "

and the proper motivation for claim 30 should have been:

"It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the Li teachings of Crisan et al. to include a workflow execution manager residing in memory for managing execution of a workflow that is repeatedly executed on the data of the database, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format because an output of one operation could be useful as an input for another. "

As to appellants' argument regarding claim 32 that Li in view of Crisan et al. does not teach "an interface for specifying a single multi-analysis functional module used to execute the plurality of functional modules, wherein the workflow is defined by plurality of steps, each step being an executable function that operates on input from a previous step and produces output for a subsequent step and each having a defined input format and output format" is not found persuasive.

The description of the limitations can be reasonable be compared to over all workflow procedure as described by Crisan et al. in paragraph 0130, lines 4-5. Appellants' note that the multi-analysis functional module is a process invoked as part of a workflow, not the workflow itself, is not relevant since it is not part of the claim language.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

Tomasz Ponikiewski Tomasz Ponikiewski

Conferees:

Eddie Lee EDDIE C. LEE
SUPERVISORY PATENT EXAMINER

Jeffrey Gaffin JEFFREY GAFFIN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100